

Our Docket No.: 42P7327



IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of:

Yeo

Application No.: 09/336,530

Filed: June 18, 1999

For: Systems and Methods for Fast Random  
Access and Backward Playback of Video  
Using Decoded Frame Cache

) Examiner: Onuaku, Christopher

) **Art Group: 2616**

APPEAL BRIEF  
IN SUPPORT OF APPELLANT'S APPEAL  
TO THE BOARD OF PATENT APPEALS AND INTERFERENCES

Sir:

Applicant (hereinafter "Appellant") hereby submits this Brief in support of its appeal from a final decision by the Examiner, mailed April 7, 2005, in the above-referenced Application. Appellant respectfully requests consideration of this appeal by the Board of Patent Appeals and Interferences (hereinafter "Board") for allowance of the above-captioned patent application.

An oral hearing is not desired.

08/15/2005 HVUONG1 00000085 09336530

01 FC:1402

500.00 OP

## **TABLE OF CONTENTS**

I.	REAL PARTY IN INTEREST	3
II.	RELATED APPEALS AND INTERFERENCES	3
III.	STATUS OF THE CLAIMS	3
IV.	STATUS OF AMENDMENTS	4
V.	SUMMARY OF THE CLAIMED SUBJECT MATTER	4
VI.	GROUND OF REJECTION TO BE REVIEWED ON APPEAL	7
VII.	ARGUMENT	8
VIII.	CONCLUSION	19
IX.	APPENDIX OF CLAIMS	i

**I. REAL PARTY IN INTEREST**

The invention is assigned to Intel Corporation of 2200 Mission College Boulevard, Santa Clara, California 95052.

**II. RELATED APPEALS AND INTERFERENCES**

To the best of Appellant's knowledge, there are no appeals or interferences related to the present appeal that will directly affect, be directly affected by, or have a bearing on the Board's decision.

**III. STATUS OF THE CLAIMS**

Claims 1-38 are currently pending in the above-referenced application. In the Final Office Action mailed April 7, 2005 (hereinafter "Final Office Action"), claims 1-8, 10-13, 15-21, and 23-38 stand rejected under U.S.C. §102(e) as being anticipated by *Toebe* VIII et al. (U.S. Patent No. 5,959,690) ("*Toebe*"). Claims 9 and 22 stand rejected under 35 U.S.C. §103(a) as being unpatentable over *Toebe* in view of *Proctor* et al. (U.S. Patent No. 6,072,830) ("*Proctor*"). In addition, claim 14 stands rejected under 35 U.S.C. §103(a) as being unpatentable over *Toebe*. Claims 1-38 are being appealed.

#### **IV. STATUS OF AMENDMENTS**

Claims 1-38 are currently pending in the subject application. These claims were finally rejected in the Final Office Action mailed April 7, 2005. The Examiner confirmed the final rejection of these claims in an Advisory Action mailed July 19, 2005 (hereinafter "Advisory Action").

In response to the Final Office Action mailed on April 7, 2005, rejecting claims 1-8, 10-13, 15-21, and 23-38 under 35 U.S.C. §102(e) and claims 9, 14, and 22 under 35 U.S.C. §103(a), Appellant filed a Response After Final pursuant to 37 C.F.R. § 1.116 on June 7, 2005. No amendments were presented. Subsequently, the Advisory Action maintaining all rejections in the Final Office Action was mailed on July 19, 2005. In response, Appellant filed a Notice of Appeal on July 7, 2005. A copy of all claims on appeal is attached hereto as an Appendix of Claims.

Appellant respectfully traverses each of these grounds of rejection.

#### **V. SUMMARY OF THE CLAIMED SUBJECT MATTER**

According to one embodiment, a method of processing a video stream is described in independent claim 1. The method includes detecting a request to randomly access a particular frame, maintaining a list of frame dependencies identifying at least a set of frames required to decode the particular frame, and determining based at least in part on the list of frame dependencies whether a decoded version of the particular frame is in a decoded frame cache, said cache configured to store an arbitrary number of previously decoded frames. (Specification at pg. 5, ll. 11-16; pg. 6, ll. 21-24; pg. 9, ln. 7 of pseudo-code listing; pg. 7, ll. 27-30.) The method further includes, if the decoded

version of the particular frame is not in the decoded frame cache and if the particular frame has a frame dependency, determining a frame dependency for the particular frame, determining which of the frames in the frame dependency are in the decoded frame cache, decoding any frame in the frame dependency that is not in the decoded frame cache and placing it in the decoded frame cache, and using at least one of the decoded frames in the frame dependency to decode the particular frame to create a decoded version of the particular frame. (Pg. 9, pseudo-code listing starting at ln. 17, ll. 1-15 of pseudo-code listing.)

Further embodiments include an article of independent claim 15 comprising a computer readable medium having instructions thereon which when executed cause a computer to perform the method described above with respect to claim 1. In addition, claim 26 describes a computer system including a processor and video processing circuitry, a display, and memory including instructions which when executed cause the processor and video processing circuitry to perform the method described above with respect to claim 1. (See also Figs 1 & 2; pg. 4, ll. 15-22.)

In a further embodiment, a method for randomly accessing a first frame of a video stream is described in independent claim 27. The method includes maintaining a list of frame dependencies identifying at least a set of frames required to decode the first frame, determining a decoding of the first frame is not in a decoded frame cache, said cache configured to store an arbitrary number of previously decoded frames, determining, based at least in part on the list of frame dependencies, a first frame dependency for the first frame comprising frames required to decode the first frame, decoding at least one of the frames of the frame dependency not present in the decoded frame cache, and placing it in the decoded frame cache, and decoding the first frame using at least one of the decoded

frames in the decoded frame cache. (Pg. 6, ll. 21-24; pg. 7, ll. 27-30; pg. 9-10, pseudo-code listing starting at pg. 9 ln. 17, ll. 1-22 of pseudo-code listing.)

Further embodiments include an article of claim 33 comprising a machine-accessible media having associated data for randomly accessing a first frame of a video stream, wherein the data, when accessed, results in the machine performing the method of claim 27.

In still a further embodiment, a method of caching decoded frames of a video in a decoded frame cache configured to store an arbitrary number of previously decoded frames is described in independent claim 37. The method includes maintaining a list of frame dependencies identifying at least a set of frames required to decode a particular frame of the video, determining based at least in part on the list of frame dependencies that a decoded version of the particular frame is not in the decoded frame cache, and determining if the particular frame has a frame dependency. (Pg. 6, ll. 21-24, pg. 9, ln. 7 of pseudo-code listing.) The method further includes, if a particular frame has a frame dependency, determining a frame dependency for the particular frame, determining which of the frames in the frame dependency are in the decoded frame cache, decoding any frame in the frame dependency that is not in the decoded frame cache and placing it in the decoded frame cache, and using at least one of the decoded frames in the frame dependency to decode the particular frame to create a decoded version of the particular frame. (Pg. 9, pseudo-code listing starting at ln. 17, ll. 9-14 of pseudo-code listing.)

**VI. GROUND OF REJECTION TO BE REVIEWED ON APPEAL**

Claims 1-8, 10-13, 15-21, and 23-38 stand rejected under U.S.C. §102(e) as being anticipated by Toebe VIII et al. (U.S. Patent No. 5,959,690) ("*Toebe*").

Claims 9 and 22 stand rejected under 35 U.S.C. §103(a) as being unpatentable over *Toebe* in view of Proctor et al. (U.S. Patent No. 6,072,830) ("*Proctor*").

Claim 14 stands rejected under 35 U.S.C. §103(a) as being unpatentable over *Toebe*.

## VII. ARGUMENT

### 1. **THE PENDING CLAIMS 1-8, 10-13, 15-21 AND 23-38 WERE IMPROPERLY REJECTED UNDER 35 U.S.C. § 102(e) BECAUSE *TOEBES* DOES NOT DISCLOSE OR SUGGEST EACH AND EVERY FEATURE OF THE PENDING CLAIMS**

Appellant respectfully submits that *Toebe*s fails to disclose or suggest the claimed invention for the reasons set forth below. In order to find anticipation under 35 U.S.C. §102, the prior art reference must teach each and every aspect of the claimed invention either explicitly or impliedly. MPEP §706.02(IV).

#### **(A) Claims 1-26 and 37-38 were improperly rejected because *Toebe*s does not disclose or suggest (1) determining which frames of a frame dependency are in a decoded frame cache and decoding any frame in the frame dependency that is not in the decoded frame cache and placing it in the decoded frame cache and (2) a decoded frame cache configured to store an arbitrary number of previously decoded frames**

Claims 1-26 and 37-38 recite an element that is not disclosed in *Toebe*s. For example, Appellant's independent claim 1 recites the following:

- A method of processing a video stream, comprising:
- (a) detecting a request to randomly access a particular frame;
  - (b) maintaining a list of frame dependencies identifying at least a set of frames required to decode the particular frame; and
  - (c) determining based at least in part on the list of frame dependencies whether a decoded version of the particular frame is in a decoded frame cache, said cache configured to store an arbitrary number of previously decoded frames, and if it is not and if the particular frame has a frame dependency:
    - (i) determining a frame dependency for the particular frame;
    - (ii) determining which of the frames in the frame dependency are in the decoded frame cache;
    - (iii) decoding any frame in the frame dependency that is not in the decoded frame cache and placing it in the decoded frame cache; and
    - (iv) using at least one of the decoded frames in the frame dependency to decode the particular frame to create a decoded version of the particular frame.



Appellant's independent claims 15, 26, and 37 recite similar features to those of claim 1.

*Toebe*s discloses a method for providing in a personal computing system random frame accurate access to an MPEG video stream at any frame. The central component of *Toebe*s is a MPEG streamer. (*Toebe*s at col. 8, ll. 51-52.) The MPEG streamer constructs a continuous virtual MPEG stream out of a source MPEG stream to send to a MPEG decoder of an MPEG player. (Col. 9, ll. 6-8.) *Toebe*s utilizes the MPEG streamer's ability to construct a derived stream from various components and to manipulate the derived stream in order to accomplish frame accurate access, reverse playback, and other special effects. *Toebe*s accomplishes the above tasks by placing the MPEG player in the proper state to display an arbitrary "target" frame. (Col. 11, ll. 54-60.) The MPEG player is placed in the proper state by way of the MPEG streamer determining the list of frames needed to decode the arbitrary "target" frame and then constructing a virtual MPEG stream out of those required frames. This virtual stream is sent to the MPEG player to reproduce the correct frame. (Col. 12, ln. 64-col. 13, ln.1.)

*Toebe*s further discloses utilizing two separate buffers for decoding in the MPEG player, a past buffer and a future buffer. These buffers receive the virtual stream sent from the MPEG streamer. Each of these buffers is capable of holding one frame at a time. The frames in these buffers are continuously replaced by other reference frames in the virtual stream during the course of decoding a group of pictures (GOP). (Col. 4, lines 21-33 & col. 12, ll. 42-50.)

- (i) *Toebe*s does not disclose or suggest determining which frames of a frame dependency are in a decoded frame cache and decoding any frame in the frame dependency that is not in the decoded frame cache and placing it in the decoded frame cache

Appellant submits that *Toebe*s does not disclose or suggest determining which frames of a frame dependency are in a decoded frame cache, and decoding any frame in the frame dependency that is not in the decoded frame cache and placing it in the decoded frame cache, as disclosed by independent claims 1, 15, 26, and 37. However, the Examiner maintains that *Toebe*s discloses such a feature. For instance, the Examiner asserts:

[T]his limitation is clearly disclosed in col. 4, lines 21-45 and col. 12 lines 51-64, wherein *Toebe*s, VIII et al disclose that in order to provide specific access to an arbitrary target frame other than an I frame, the necessary reference frames must be properly parsed and residing in the appropriate buffer (past and future buffers) in order for the player to be in the proper state to parse (decode) the target frame. And if the appropriate frames are not parsed and read into the appropriate buffers, the target P or B picture cannot be parsed and displayed correctly.

Advisory Action at page 2, Response to Arguments section.

Appellant submits that the portions of *Toebe*s cited by the Examiner are merely reciting a statement of fact, and are not disclosing the features of claim 1, namely determining which frames of a frame dependency are in a decoded frame cache, and decoding any frame in the frame dependency that is not in the decoded frame cache and placing it in the decoded frame cache. Col. 4, lines 21-45 and col. 12, lines 51-64 of *Toebe*s does disclose what the Examiner has recited, namely that in order to accurately provide frame specific access to an arbitrary target frame, necessary reference frames must be properly parsed and in the appropriate buffers, otherwise a P or B picture will not be parsed and displayed correctly. More simply put, the cited portion of *Toebe*s makes

clear that if the wrong frames are in the past and future buffers, than an incorrect frame will be parsed and displayed.

However, the cited portions of *Toebe*s do not, as the Examiner concludes, disclose the features of claim 1. Even though appropriate frames must be in the proper buffers for accurate parsing and displaying, this does not lead to the conclusion that a determination should be made of which frames of a frame dependency are in a decoded frame cache. Nor does it then lead to decoding any frame in the frame dependency that is not in the decoded frame cache and placing that decoded frame in the decoded frame cache. Appellant can find no disclosure or suggestion of such features anywhere in the factual statement of *Toebe*s cited by the Examiner.

The Examiner further states, in response to Appellant's argument that *Toebe*s does not disclose or suggest the above-recited features of claim 1:

[I]n Fig. 8; col. 13, line 35 to col. 15, line 13, *Toebe*s, VIII et al clearly discloses examples to illustrate frame specific access to B, P and I frames. For example, in this process, the identity of the target frame is determined. Once the identity of the target frame is determined, the location in the video bitstream of the reference frames is determined, and the frame dependencies of the target frame are determined. The frame dependencies of the target frame must be properly parsed and resident in the appropriate buffers for the target frame to be parsed (decoded).

Advisory Action at page 3, Response to Arguments section.

However, upon review of col. 13, line 35 to col. 15, line 13 of *Toebe*s, at no point does *Toebe*s disclose examining if a frame is already located in the buffer (i.e., determining which frames of a frame dependency are in a decoded frame cache). *Toebe*s utilizes a MPEG streamer to create a virtual stream according to the necessary order of frames for correct parsing of an arbitrarily accessed frame. This virtual stream is fed into the MPEG player, which contains the past and future buffers, and is processed like an

ordinary forward-playing stream would be processed by automatically feeding the frames into the past and future buffers based on the order they are received. *Toebe's* relies on the set order of frames in the virtual bitstream and automatically places frames in the past and future buffers based on that order. (*Toebe's* at col. 12, ln. 64-col. 13, ln. 1; col. 12, ll. 42-50; col. 13, ll. 58-61.)

For example, with each I, P, or B frame there is a process by which they are parsed, and either displayed or placed into a buffer. (See, e.g., *Toebe's* at Fig. 8.) This process is done without regard to the state of the buffers. For example, *Toebe's* states that "we do not need to be concerned with the state of the MPEG player's past buffer. . . . the process of the invention merely assures that the I frame is parsed into the future buffer and the streamer/player is poised to parse the next reference frame upon enablement of the display and resumption of normal play." (Col. 16, lines 29-37.) Consequently, reference frames in the buffers are continuously replaced by other reference frames in the bitstream. (Col. 4, lines 21-33; col. 12, lines 42-50). If it was desired to return to a dependent frame ("P" or "B" frame) that has already been displayed, the decoding process would have to be repeated because the required dependency reference frames would no longer be in the buffers.

Figure 9 of *Toebe's* and its corresponding explanation at col. 17, line 35 to col. 18, line 19 exemplify this point. Specifically, Figure 9 discloses the reverse play of an MPEG sequence. At step 2 of Figure 9, the MPEG player is placed in suppression mode to parse pictures 0, 3, and 6 so that picture 9 can be parsed. Yet, at step 6, figures 0, 3, and 6 must be parsed again to place the player in the proper state to decode pictures 5 and 4. If figures 0, 3, and 6 had been placed in a decoded frame cache the first time they

were decoded at step 2, then, at step 6, it would have been determined that these frames were already in the decoded frame cache and would not have to be parsed again.

In comparison, the embodiments of the present application allows reference frames, once decoded, to remain in a decoded frame cache. These reference frames are not arbitrarily bumped by other reference frames that arrive later in a bitstream. Accordingly, the decoded frame cache of claim 1 can be referenced to determine whether a frame is already decoded so that the frame can be used and does not have to be decoded again. There is no disclosure or suggestion of such a feature in *Toebe*s. Furthermore, because *Toebe*s does not disclose or suggest determining whether a reference frame is already in a decoded frame cache, it cannot disclose making a decision whether or not to decode a frame based on that determination.

Therefore, Appellant submits that *Toebe*s does not disclose or suggest determining which frames of a frame dependency are in a decoded frame cache, and decoding any frame in the frame dependency that is not in the decoded frame cache and placing it in the decoded frame cache. Therefore, independent claims 1, 15, 26, and 37 are patentable over *Toebe*s.

**(ii) *Toebe*s does not disclose or suggest a decoded frame cache configured to store an arbitrary number of previously decoded frames**

Additionally, Appellant submits that *Toebe*s does not disclose or suggest a decoded frame cache configured to store an arbitrary number of previously decoded frames, as disclosed by claims 1, 15, 26, and 37. *Toebe*s discloses the use of a future buffer and a past buffer. These buffers are used to constantly keep the last two reference frames available for use. Only one frame at a time is kept in each buffer. (See *Toebe*s at col. 4, ll. 21-33.) Appellant can find no disclosure or suggestion in *Toebe*s of a decoded frame cache configured to store an arbitrary number of previously decoded frames. As

mentioned, the past and future buffers of *Toebe*s are configured to store a single frame each. This is not the same as being configured to store an *arbitrary number* of previously decoded frames.

Therefore, Appellant submits that *Toebe*s does not disclose or suggest a decoded frame cache configured to store an arbitrary number of previously decoded frames. As a result, independent claims 1, 15, 26, and 37 are patentable over *Toebe*s.

For the foregoing reasons, Appellant submits that the Examiner has failed to search and find a printed publication or patent that discloses the claimed invention as set forth in MPEP § 706.02(a).

Claims 2-14, 16-25, and 38 depend from claims 1, 15, and 37, respectively. Given that dependent claims necessarily include the limitations of the claims from which they depend, Appellant submits that the invention as claimed in claims 2-14, 16-25, and 38 is similarly patentable over *Toebe*s.

Thus, the Examiner erred in rejecting claims 1-26 under U.S.C. § 102(e).

**(B) Claims 27-36 were improperly rejected because *Toebe*s does not disclose or suggest determining a decoding of a first frame is not in a decoded frame cache**

Claims 27-36 are not anticipated under 35 U.S.C. §102(e) for the same reasons as given above with respect to claims 1-26 and 37-38, and further due to the additional limitation of determining a decoding of a first frame is not in a decoded frame cache.

Appellant's arguments made above with respect to claims 1-26 apply equally to claims 27-36 and are incorporated herein by reference. With respect to the determining a

decoding of a first frame is not in a decoded frame cache, Appellant's claim 27 recites the following:

A method for randomly accessing a first frame of a video stream, comprising:  
maintaining a list of frame dependencies identifying at least a set of frames required to decode the first frame;  
determining a decoding of the first frame is not in a decoded frame cache, said cache configured to store an arbitrary number of previously decoded frames;  
determining, based at least in part on the list of frame dependencies, a first frame dependency for the first frame comprising frames required to decode the first frame;  
decoding at least one of the frames of the frame dependency not present in the decoded frame cache, and placing it in the decoded frame cache; and  
decoding the first frame using at least one of the decoded frames in the decoded frame cache.

Appellant submits that nowhere in *Toebe's* is there disclosed determining a decoding of a first frame is not in a decoded frame cache. As discussed above with respect to claim 1-26 and 37-38, *Toebe's* does not disclose determining which frames of a frame dependency are in a decoded frame cache. Similarly, Appellant can find no disclose or suggestion in *Toebe's* of determining that a decoding of a first frame is not in a decoded frame cache. The virtual bitstream of *Toebe's* is fed into the MPEG player and automatically placed into the past and future buffers according to the bitstream order. *Toebe's* does not disclose examining the past and future buffers to determine that a frame is not already in those buffers. Accordingly, claim 27 is patentable over *Toebe's*.

Claims 28-36 depend from claim 27 and include additional limitations. Therefore, the invention as claimed in claims 28-26 are similarly patentable over *Toebe's*.

For the forgoing reasons, Appellant submits that the Examiner has failed to search and find a printed publication or patent that discloses the claimed invention as set forth in MPEP § 706.02(a).

Thus, the Examiner erred in rejecting claims 27-36 under 35 U.S.C. §102(e).

**2. THE PENDING CLAIMS 9 AND 22 WERE IMPROPERLY REJECTED UNDER 35 U.S.C. § 103(a) BECAUSE ANY COMBINATION OF *TOEBES* AND *PROCTOR* DO NOT DISCLOSE OR SUGGEST EACH AND EVERY FEATURE OF THE PENDING CLAIMS**

Appellant respectfully submits that *Toebe*s in view of *Proctor* fails to disclose or suggest the claimed invention for the reasons set forth below. As the Honorable Board is well aware, in order to establish a *prima facie* case of obviousness:

First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations.” (Emphasis added). *In re Vaech*, 947 F.2d 488, 20 USPQ2d 1438 (Fed. Cir. 1991). Manual of Patent Examining Procedure (MPEP), 8<sup>th</sup> Edition, Revision 2, May 2004, §2143.

**(A) Claims 9 and 22 were improperly rejected because *Toebe*s in view of *Proctor* does not disclose or suggest (1) determining which frames of a frame dependency are in a decoded frame cache and decoding any frame in the frame dependency that is not in the decoded frame cache and placing it in the decoded frame cache and (2) a decoded frame cache configured to store an arbitrary number of previously decoded frames**

Claims 9 and 22 are not obvious in view of *Toebe*s and *Proctor* under 35 U.S.C. §103(a).

As discussed above, nowhere does *Toebe*s teach or suggest each and every element of the Appellant’s independent claims. For example, *Toebe*s does not teach determining which frames of a frame dependency are in a decoded frame cache and



decoding any frame in the frame dependency that is not in the decoded frame cache and placing it in the decoded frame cache, and does not teach a decoded frame cache configured to store an arbitrary number of previously decoded frames.

With respect to claim 9, which depends from independent claim 1, the Examiner cites *Proctor* for teaching replacing decoded frames in a decoded frame cache according to a least recently used policy. (See Final Office Action at pg. 12, point 5.) However, since *Toebe*s fails to disclose many of the elements required by the Appellant's independent claims, including claim 1, and since *Proctor* fails to disclose, teach and/or suggest those elements missing from *Toebe*s, the combination of *Toebe*s and *Proctor* fails to teach or suggest each and every element of the Appellant's invention as embodied in the claims. Consequently, the Examiner has not established a prima facie case of obviousness, and the Examiner's rejection of claim 9 under 35 U.S.C. §103(a) as being obvious over the combination of *Toebe*s and *Proctor* should be reversed.

With respect to claims 22, the Examiner also cites *Proctor* for teaching replacing decoded frames in a decoded frame cache according to a least recently used policy. (Id.) However, since *Toebe*s fails to disclose many of the elements required by the Appellant's independent claims, including claim 15, and since *Proctor* fails to disclose, teach and/or suggest those elements missing from *Toebe*s, the combination of *Toebe*s and *Proctor* fails to teach or suggest each and every element of the Appellant's invention as embodied in the claims. Consequently, the Examiner has not established a prima facie case of obviousness, and the Examiner's rejection of claim 22 under 35 U.S.C. §103(a) as being obvious over the combination of *Toebe*s and *Proctor* should be reversed.

3. **THE PENDING CLAIM 14 WAS IMPROPERLY REJECTED UNDER 35 U.S.C. § 103(a) BECAUSE ANY COMBINATION OF *TOEBES* AND ORDINARY SKILL IN THE ART DO NOT DISCLOSE OR SUGGEST EACH AND EVERY FEATURE OF THE PENDING CLAIMS**

Appellant respectfully submits that *Toebe*s in view of AAPA fails to disclose or suggest the claimed invention for the reasons set forth below.

(A) **Claim 14 was improperly rejected because *Toebe*s in view of Ordinary Skill in the Art does not disclose or suggest (1) determining which frames of a frame dependency are in a decoded frame cache and decoding any frame in the frame dependency that is not in the decoded frame cache and placing it in the decoded frame cache and (2) a decoded frame cache configured to store an arbitrary number of previously decoded frames**

Claim 14 is not obvious in view of *Toebe*s under 35 U.S.C. §103(a). Claim 14 depends from independent claim 1 and necessarily includes each of its features. As discussed above, nowhere does *Toebe*s teach or suggest each and every element of the Appellant's independent claim 1. For example, *Toebe*s does not teach determining which frames of a frame dependency are in a decoded frame cache and decoding any frame in the frame dependency that is not in the decoded frame cache and placing it in the decoded frame cache, and does not teach a decoded frame cache configured to store an arbitrary number of previously decoded frames.

With respect to claim 14, the Examiner states that it is well known for decoded frame cache to include a section of main memory. However, since *Toebe*s fails to disclose many of the elements required by the Appellant's independent claim 1, and since ordinary skill in the art fails to disclose, teach and/or suggest those elements missing from *Toebe*s, the combination of *Toebe*s and ordinary skill in the art fails to teach or suggest each and every element of the Appellant's invention as embodied in the claims. Consequently, the Examiner has not established a prima facie case of obviousness, and

the Examiner's rejection of claim 14 under 35 U.S.C. §103(a) as being obvious over *Toebe*s should be reversed.

### VIII. CONCLUSION

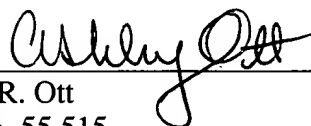
Careful review of the Examiner's rejections shows that the Examiner has failed to provide any reference, or combination of references of the prior art that shows all of the elements of each appealed claim. Therefore, Appellant respectfully submits that all appealed claims in this application are patentable and were improperly rejected by the Examiner during prosecution before the United States Patent and Trademark Office. Appellant respectfully requests that the Board of Patent Appeals and Interferences overrule the Examiner and direct allowance of the rejected claims.

This brief is submitted with a check for \$500.00 to cover the appeal fee for one other than a small entity as specified in 37 C.F.R. § 1.17(c). Please charge any shortages and credit any overcharges to our Deposit Account No. 02-2666.


Respectfully submitted,

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP

Dated: August 9, 2005

  
\_\_\_\_\_  
Ashley R. Ott  
Reg. No. 55,515

12400 Wilshire Boulevard  
Seventh Floor  
Los Angeles, CA. 90025-1026  
(408) 720-8598

FIRST CLASS CERTIFICATE OF MAILING	
I hereby certify that I am causing the above-referenced correspondence to be deposited with the United States Postal Service as first class mail with sufficient postage on the date indicated below and that this paper or fee has been addressed to the Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450	
Date of Deposit:	<u>August 9, 2005</u>
Name of Person Mailing Correspondence:	<u>Leah Schwenke</u>
 Signature	<u>August 9, 2005</u> Date

**IX. APPENDIX OF CLAIMS (37 C.F.R. § 41.37(c)(1)(viii))**



The claims on appeal read as follows:

1. A method of processing a video stream, comprising:
  - (a) detecting a request to randomly access a particular frame;
  - (b) maintaining a list of frame dependencies identifying at least a set of frames required to decode the particular frame; and
  - (c) determining based at least in part on the list of frame dependencies whether a decoded version of the particular frame is in a decoded frame cache, said cache configured to store an arbitrary number of previously decoded frames, and if it is not and if the particular frame has a frame dependency:
    - (i) determining a frame dependency for the particular frame;
    - (ii) determining which of the frames in the frame dependency are in the decoded frame cache;
    - (iii) decoding any frame in the frame dependency that is not in the decoded frame cache and placing it in the decoded frame cache; and
    - (iv) using at least one of the decoded frames in the frame dependency to decode the particular frame to create a decoded version of the particular frame.
2. The method of claim 1, wherein the request to playback a particular frame is part of a request to perform frame-by-frame backward playback and part (c) is performed for successively earlier frames with respect to the particular frame as part of the frame-by-frame backward playback.
3. The method of claim 1, wherein part (i) is performed whether or not it is determined that a decoded version of a particular frame is in the decoded frame cache without part (iv) being performed.
4. The method of claim 1, wherein the particular frame may be an I, P, or B frame of MPEG compressed video.
5. The method of claim 1, wherein the frame dependency is an immediate frame dependency.

6. The method of claim 5, wherein the at least some of the decoded frames referred to in part (iv) are those frames in the immediate dependency.
7. The method of claim 5, wherein part (c) includes recursion where frames in the immediate frame dependency of the frame of interest are not in the decoded frame cache.
8. The method of claim 1, wherein part (c) includes a loop with a terminating condition that all frames on which the particular frame is dependent have been decoded.
9. The method of claim 1, wherein decoded frames are replaced in the decoded frame cache according to a least recently used policy.
10. The method of claim 1, wherein an index is used to represent each frame in the frame dependency.
11. The method of claim 1, wherein the frame dependency is determined through a look-up table.
12. The method of claim 11, wherein the frame dependency is determined through successive uses of a look-up table.
13. The method of claim 1, wherein the decoded frame cache includes a data structure.
14. The method of claim 1, wherein the decoded frame cache includes a section of main memory.
15. An article comprising:
  - a computer readable medium having instructions thereon which when executed cause a computer to:
    - (a) detect a request to randomly access a particular frame; and
    - (b) maintaining a list of frame dependencies identifying at least a set of frames required to decode the particular frame;
    - (c) determine base at least in part on the list of frame dependencies whether a decoded version of the particular frame is in a decoded frame cache, said cache

configured to store an arbitrary number of previously decoded frames, and if it is not and if the particular frame has a frame dependency:

- (i) determine a frame dependency for the particular frame;
- (ii) determine which of the frames in the frame dependency are in the decoded frame cache;
- (iii) decode any frame in the frame dependency that is not in the decoded frame cache and place it in the decoded frame cache; and
- (iv) use at least and of the decoded frames in the frame dependency to decode the particular frame to create a decoded version of the particular frame.

16. The article of claim 15, wherein the request to playback a particular frame is part of a request to perform frame-by-frame backward playback and part (c) is performed for successively earlier frames with respect to the particular frame as part of the frame-by-frame backward playback.

17. The article of claim 15, wherein part (i) is performed whether or not it is determined that a decoded version of a particular frame is in the decoded frame cache without part (iv) being performed.

18. The article of claim 15, wherein the frame dependency is an immediate frame dependency.

19. The article of claim 18, wherein the at least some of the decoded frames referred to in part (iv) are those frames in the immediate dependency.

20. The article of claim 18, wherein part (c) includes recursion where frames in the immediate frame dependency of the frame of interest are not in the decoded frame cache.

21. The article of claim 15, wherein part (c) includes a loop with a terminating condition that all frames on which the particular frame is dependent have been decoded.

22. The article of claim 15, wherein decoded frames are replaced in the decoded frame cache according to a least recently used policy.

23. The article of claim 15, wherein an index is used to represent each frame in the frame dependency.

24. The article of claim 15, wherein the frame dependency is determined through a look-up table.

25. The article of claim 24, wherein the frame dependency is determined through successive uses of a look-up table.

26. A computer system including:

a processor and video processing circuitry;

a display; and

memory including instructions which when executed cause the processor and video processing circuitry to:

(a) detect a request to randomly access a particular frame; and

(b) maintain a list of frame dependencies identifying at least a set of frames required to decode the particular frame;

(c) determine whether a decoded version of the particular frame is in a decoded frame cache, said cache configured to store an arbitrary number of previously decoded frames, and if it is not and if the particular frame has a frame dependency:

(i) determine a frame dependency for the particular frame;

(ii) determine which of the frames in the frame dependency are in the decoded frame cache;

(iii) decode any frame in the frame dependency that is not in the decoded frame cache and place it in the decoded frame cache; and

(iv) use at least and of the decoded frames in the frame dependency to decode the particular frame to create a decoded version of the particular frame.

(d) provide the decoded version of the particular frame for displaying on the display.

27. A method for randomly accessing a first frame of a video stream, comprising:  
maintaining a list of frame dependencies identifying at least a set of frames required to decode the first frame;

determining a decoding of the first frame is not in a decoded frame cache, said cache configured to store an arbitrary number of previously decoded frames;

determining, based at least in part on the list of frame dependencies, a first frame dependency for the first frame comprising frames required to decode the first frame;

decoding at least one of the frames of the frame dependency not present in the decoded frame cache, and placing it in the decoded frame cache; and

decoding the first frame using at least one of the decoded frames in the decoded frame cache.

28. The method of claim 27, further comprising:

decoding each frame of the frame dependency not present in the decoded frame cache, and placing them in the decoded frame cache.

29. The method of claim 27, further comprising:

recursively decoding the second frame of the frame dependency.

30. A method according to claim 27 for reverse playback of frames of the video stream, comprising:

determining a second frame is not in the decoded frame cache, the second frame following the first frame in the video stream;

determining a second frame dependency for the second frame comprising frames required to decode the second frame;

decoding at least one of the frames of the frame dependency not present in the decoded frame cache, and placing it in the decoded frame cache; and

decoding the second frame using at least one of the decoded frames in the decoded frame cache.

31. The method of claim 30, further comprising:

playing the second frame and then the first frame.

32. The method of claim 30, wherein the second frame is an immediately following frame of the first frame.

33. An article comprising a machine-accessible media having associated data for randomly accessing a first frame of a video stream, wherein the data, when accessed, results in a machine performing:

maintaining a list of frame dependencies identifying at least a set of frames required to decode the first frame;



determining a decoding of the first frame is not in a decoded frame cache, said cache configured to store an arbitrary number of previously decoded frames;

determining, based at least in part on the list of frame dependencies, a first frame dependency for the first frame comprising frames required to decode the first frame;

decoding at least one of the frames of the frame dependency not present in the decoded frame cache, and placing it in the decoded frame cache; and

decoding the first frame using at least one of the decoded frames in the decoded frame cache.

34. The article of claim 33 wherein the machine-accessible media further includes data, when accessed, results in the machine performing:

decoding each frame of the frame dependency not present in the decoded frame cache, and placing them in the decoded frame cache.

35. The article of claim 33 wherein the machine-accessible media further includes data, when accessed, results in the machine performing:

recursively decoding the second frame of the frame dependency.

36. The article of claim 33 wherein the machine-accessible media further includes data for reverse playback of frames of the video stream, when accessed, results in the machine performing:

determining a second frame is not in the decoded frame cache, the second frame following the first frame in the video stream;

determining a second frame dependency for the second frame comprising frames required to decode the second frame;

decoding at least one of the frames of the frame dependency not present in the decoded frame cache, and placing it in the decoded frame cache; and

decoding the second frame using at least one of the decoded frames in the decoded frame cache.

37. A method of caching decoded frames of a video in a decoded frame cache configured to store an arbitrary number of previously decoded frames, comprising:

maintaining a list of frame dependencies identifying at least a set of frames required to decode a particular frame of the video;

determining based at least in part on the list of frame dependencies that a decoded version of the particular frame is not in the decoded frame cache; and  
determining if the particular frame has a frame dependency, and if so:  
determining a frame dependency for the particular frame,  
determining which of the frames in the frame dependency are in the decoded frame cache,  
decoding any frame in the frame dependency that is not in the decoded frame cache and placing it in the decoded frame cache, and  
using at least one of the decoded frames in the frame dependency to decode the particular frame to create a decoded version of the particular frame.

38. The method of claim 37, further comprising:  
detecting a request to randomly access the particular frame;  
wherein the request to playback the particular frame is part of a request to perform frame-by-frame backward playback.

AUG 15 2005

# FEE TRANSMITTAL for FY 2005

Patent fees are subject to annual revision.

Complete if Known

Application Number	09/336,530
Filing Date	June 18, 1999
First Named Inventor	Boon-Lock Yeo
Examiner Name	Christopher O. Onuaku
Art Unit	2616
Attorney Docket No.	42390P7327

☐ Applicant claims small entity status. See 37 CFR 1.27.

TOTAL AMOUNT OF PAYMENT (\$) 500.00

## METHOD OF PAYMENT (check all that apply)

☒ Check ☐ Credit card ☐ Money Order ☐ None ☐ Other (please identify):

☒ Deposit Account Deposit Account Number: 02-2666 Deposit Account Name: Blakely, Sokoloff, Taylor & Zafman LLP

For the above-identified deposit account, the Director is hereby authorized to: (check all that apply)

☐ Charge fee(s) indicated below ☐ Charge fee(s) indicated below, except for the filing fee

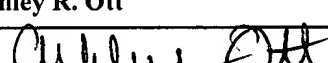
☒ Charge any additional fee(s) or underpayment of fee(s) under 37 CFR §§ 1.16, 1.17, 1.18 and 1.20. ☒ Credit any overpayments

## FEE CALCULATION

Large Entity		Small Entity		Fee Description	Fee Paid
Fee Code	Fee (\$)	Fee Code	Fee (\$)		
1051	130	2051	65	Surcharge - late filing fee or oath	
1052	50	2052	25	Surcharge - late provisional filing fee or cover sheet.	
2053	130	2053	130	Non-English specification	
1251	120	2251	60	Extension for reply within first month	
1252	450	2252	225	Extension for reply within second month	
1253	1,020	2253	510	Extension for reply within third month	
1254	1,590	2254	795	Extension for reply within fourth month	
1255	2,160	2255	1,080	Extension for reply within fifth month	
1401	500	2401	250	Notice of Appeal	
1402	500	2402	250	Filing a brief in support of an appeal	500.00
1403	1,000	2403	500	Request for oral hearing	
1451	1,510	2451	1,510	Petition to institute a public use proceeding	
1460	130	2460	130	Petitions to the Commissioner	
1807	50	1807	50	Processing fee under 37 CFR 1.17(q)	
1806	180	1806	180	Submission of Information Disclosure Stmt	
1809	790	1809	395	Filing a submission after final rejection (37 CFR § 1.129(a))	
1810	790	2810	395	For each additional invention to be examined (37 CFR § 1.129(b))	
Other fee (specify)					
				SUBTOTAL (2)	500.00

## SUBMITTED BY

Complete (if applicable)

Name (Print/Type)	Ashley R. Ott	Registration No. (Attorney/Agent)	55,515	Telephone	(303) 740-1980
Signature		Date	08/09/05		